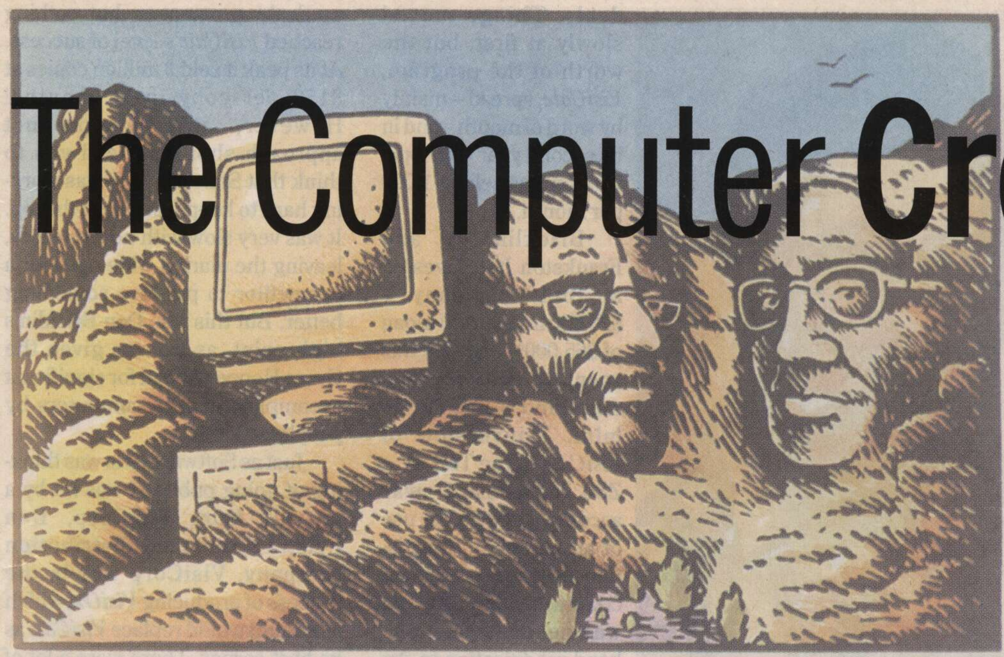
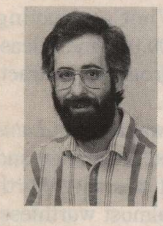
The Computer Creators



The spreadsheet helped bring the computer into everyday use. Mike James looks at the work of its inventor, Dan Bricklin

COMPUTERS WERE ONCE the sole preserve of programmers. If you wanted a computer to do something, be it as simple as adding two numbers together, you needed a program and hence a programmer. This was a severe problem in the early days of the personal computer because, for the computer to become truly personal, the number of computers in existence needed to overtake the number of programmers! At first the solution seemed to be to make programming easy enough for anyone to do, which provoked lots of talk about vague concepts such as 'computer literacy'. As we all now know, the solution was not to make everyone a programmer but to remove the need to program. At the time this seemed an impossibly difficult task with no obvious first step. But if anything deserves the title of 'first step to a programmerless world', it's probably the spreadsheet. And Dan Bricklin was the programmer who was in the right place at the right time to invent it. Or did it take more – a touch of genius, perhaps?

File card: Dan Bricklin



Name:	Dan Bricklin
Born:	1951
Country:	USA
Specialism:	Software
Theoretical:	None
Product:	Visicalc, TK!Solver, Dan Bricklin's Demo
Company:	Software Arts, Software Garden

Computing is such a fast-moving field that it's hard to believe the electronic spreadsheet was invented only 15 years ago. The spreadsheet seems so obvious an idea that it hardly needs thinking up. Indeed, once it was thought of by Dan Bricklin, his biggest worry was that someone else would be first to the market with a real product, having discovered the idea independently!

Dan Bricklin learned about computers in the days of mainframes and minis. He graduated from MIT (Massachusetts Institute of Technology) in 1973, specialising in electronics and computer science. After this he followed what many programmers will recognise as a typical apprenticeship. While working at MIT he had written a program for Multics, an advanced operating system, which implemented a command line calculator. Bricklin was surprised that such an advanced system lacked any way of allowing a user to work out 2+2 without writing a program. Perhaps the seed of the spreadsheet idea was sown in this early observation. After spending time working on implementing advanced language systems – APL and Lisp – Bricklin's apprenticeship took a turn for the practical when he joined the DEC typesetting group. This involved a great deal of troubleshooting for daily newspapers which were trying to keep to deadlines, which meant he met real users and saw how important the user interface really was!

After a spell writing a word processor for a 16K, 12-bit word PDP 8 minicomputer, Bricklin decided it was time to go back to school – but this time to get an MBA, which he hoped would get get him a better job. However, there was also the half-formed thought that it might be useful knowledge should he ever want to start his own business.

Back to school

The Harvard business school, where Bricklin enrolled, was famous for its case study approach. What this actually meant was that the students were expected to take data and perform long, complicated and tedious calculations. Company data was written up on lots of blackboards along with calculated values. Changing a single value usually entailed recalculating many of the results on the blackboard. Worse, a single error would propagate through the calculations and you could spend hours looking for it. As well as being a student exercise, this was also the way that large companies actually did their financial planning: using hoards of economists with yards of blackboards – a sort of chalk, blackboard and human precursor of the spreadsheet 'what-if' facility.

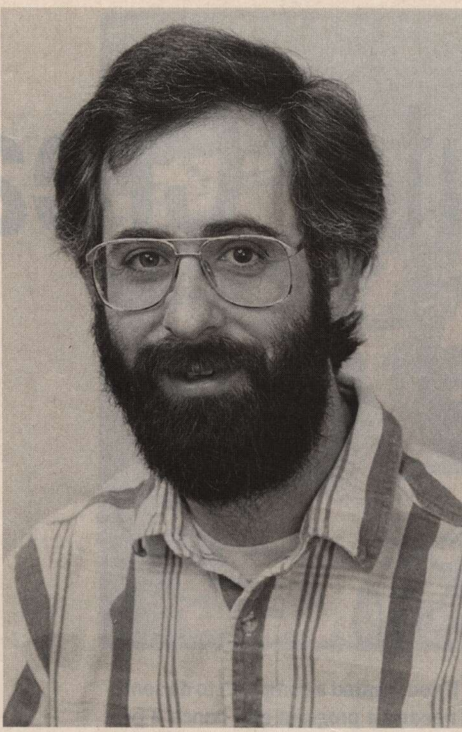
There is absolutely no question that the calculator, pencil and paper exercises focused Bricklin's mind on how to do the job with a minimum of sweat. Given that he was a good programmer, you might think that the solution to his problem would be to write Basic to solve each assignment. The trouble with this was that each assignment needed a completely new program, which in itself would take a long time to write. It was at this time that Bricklin conceived the idea of 'an electronic calculator – the word processor that would work with numbers'. At this point, Bricklin's ideas were a little hazy but they were also ambitious. He originally thought of the grid of cells that we now recognise as a spreadsheet, but he also wanted to use a mouse to point at cells and manipulate values.

The idea was a good one, but Bricklin needed a second opinion. He asked his Harvard professors what they thought of the idea. In the main they were remarkably positive, apart from one, who told ➧

him that there were lots of financial modelling programs and the market was probably saturated. But if that was the case, why then were they doing what-if analysis on blackboards? Fortunately Bricklin didn't despair, but the comment must have added some doubts to his plan to build the program we all needed.

The doubting professor at least suggested that Bricklin contacted someone who might know better – Dan Flystra, an ex-student who had a plan to start a software distribution company. Flystra told Bricklin to contact him when and if he had a product. Not discouraging, but hardly the positive response that the most important idea in computer applications deserved.

Undeterred, Bricklin decided to spend his summer vacation implementing a demo. This was written in Basic on an Apple II that Flystra made available. A few hundred lines of code animated 100 cells – if 20 seconds per recalculation can be called animation! There was no printing facility, and the 10x10 grid fitted onscreen so there was no need for scrolling, but it was recognisably an electronic spreadsheet, complete with A,B,C for the columns and 1,2,3 for the rows. At first the game paddles of the Apple II were used in place of a mouse for positioning, but eventually this was dropped in favour of the cursor keys. This demo was used to work out many of the important features of the interface – the slash to initiate a command and the automatic entering of cell references by pointing.



Dan Bricklin – without him, we would be a spreadsheet-less society. Thank goodness for Dan, eh?

Two's company

The next step was to develop an assembly language version, but this was more than Bricklin could attempt, given that he was just starting the second year of his MBA course. Enter Bob Frankston. Bricklin had met Frankston while at MIT and there had been a sort of agreement between them that one day they would start their own company. The spreadsheet idea seemed the ideal starting point: Frankston would write the code and Bricklin would oversee the design. They set up a company, Software Arts, with Bricklin as chairman and Frankston as president. As is the tradition of such early startups, Frankston worked the nightshift to save money on the timeshare service they needed to use to create the program.

At this stage more sophisticated problems arose, such as how to implement replication and absolute and relative cell references. The final version of the program had to be a compromise of features and performance, and the first assembly language version used only 25K of Ram (no, there isn't a zero missing, it really was 25K!), but it only worked with integers and could only do addition and subtraction.

"VisiCalc made both the Apple and the IBM PC a success. IBM wouldn't announce the PC until VisiCalc was ready for it"

Apple was shown the first version and failed to see any value in it. Atari thought it was good but didn't have a machine ready. Flystra, on the other hand, agreed to distribute it for a huge 64.3 percent of the revenue, falling to 50 percent for bulk deals. Things moved slowly at first, but the worth of the program, *VisiCalc*, spread – mainly by word of mouth – and in its second year the Apple version was selling 1,000 per month.

Bricklin and Frankston had invested in their company – $20,000 in cash and a loan of $65,000 – and this was spent on two programmers and a Prime 550 mini. Conversion of *VisiCalc* to the TRS-80, a Z80 processor, was accomplished by the use of an automatic translation program – a clever cross assembler. All that was needed was some hand modification to the output. Years later, they discovered that the bugs in the 6502 code from the Apple had been perfectly translated into the Z80 code.

They used the same technique to convert *VisiCalc* to the IBM PC. This ensured that they could get the program onto the machine quickly, but it also gave Lotus the chance to get in and produce a better program. A program that's translated can never be as efficient as one that's hand-crafted for a specific processor. Hence Lotus could produce something faster and with more features in the same amount of memory – but exactly how it did it is another story.

VisiCalc made both the Apple and the IBM PC a success. IBM wouldn't announce the PC until *VisiCalc* was ready for it, while people would buy a personal computer simply so they could run the program. *VisiCalc* sold the personal computer to the mass market, as at last it enabled users to work with numbers and construct models without having to write programs. The triumph of the spreadsheet continues today, with it still ranking as the number two application after word processing.

First time lucky…

But for Software Arts it was all downhill. After creating the spreadsheet, the company lost focus. Bricklin worried about producing a second product for fear of it ruining their superb track record in selling *VisiCalc*. Eventually the company did produce *TK!Solver*, an early attempt at an equation-solving program, and *SpotLight*, an early pop-up desktop manager, but nothing reached *VisiCalc*'s level of success. At its peak it sold 2 million copies at $150 per copy in 12 months! However, this one-product approach shouldn't lead you to think that Software Arts was working hard to keep *VisiCalc* at the top. It was very slow with new releases, leaving the market wide open for a competitor to produce something better. But this was Dan Bricklin's high point, as he was given the Grace Hopper Award for significant accomplishment in computers by people under 30.

Just as Software Arts was thinking of going public, it was hit by a lawsuit from VisiCorp, Dan Fylstra's renamed distribution company. VisiCorp was in the process of producing *VisiOn*, which it thought would replace *VisiCalc* as its best earner. There were a lot of arguments between Software Arts and VisiCorp, and the final split resulted in a $60 million lawsuit alleging that Software Arts had failed to enhance *VisiCalc* and produce new versions.

Software Arts hit back with a lawsuit claiming that VisiCorp had favoured its own products at the expense of *VisiCalc*. This messy lawsuit brought the progress of the company share issue to a halt. No-one wants to buy into a company when it's in the middle of litigation. Eventually the case was settled out of court, with VisiCorp paying Software Arts $500,000 and releasing the company from the contract. But it was too late.

By the time it was all over, *Lotus 1-2-3* had cornered the market and the company that was once worth $50 million was almost worthless. Lotus picked up Software Arts' products and other assets for next to nothing and thus avoided the 'look-and-feel' lawsuit that Software Arts might have won against it in the following decade! Dan Bricklin watched his company being sold off and moved on to start a small-time business called Software Garden – not rich but at least a little famous. But despite the fame, Software Garden has been a relatively low-profile operation. Its best known product, *Dan Bricklin's Demo*, was over-enthusiastically received, and there's a sense in which anything he does is looked at as a potential 'next spreadsheet'. One of Bricklin's legacies is that he has created an expectation in the PC industry that the saviour 'next spreadsheet' will soon come. I think it was a one-off event. ■